REMARKS

Claims 1-29 were originally pending. Claims 5, 8, 11, and 22 have been amended. No claims have been added. No claims have been canceled or withdrawn. Accordingly, claims 1-29 remain pending.

In view of the following remarks/arguments, withdrawal of all outstanding objections and rejections to the pending claims is respectfully requested.

Claim Objections

Claims 8 and 22 stand objected to for grammatical informalities. Claims 8 and 22 have been amended to correct the indicated informalities. In view of these amendments, withdrawal of the objections to claims 8 and 22 is respectfully requested.

35 USC §112, Second Paragraph, Rejections

Claims 5, 7, 11, 14, and 21 stand rejected under 35 USC §112, second paragraph as failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

In addressing claims 5 and 11, the Office Action ("Action") asserts that "it is unclear how 'wherein the scheduling' is meant to limit the claim. Applicant respectfully disagrees with this assertion. The phrase "wherein the scheduling" in clearly points to the gerund in respective base claims 1 or 8 of "scheduling". Thus, the phrase in question in both claims 5 and 8 particularly points out and distinctly claims the subject matter of the invention. However, since claims 5 and 8 will still particularly point out

that "the predetermined periodic rate is a millisecond", even without the rejected phrase, claims 5 and 11 have been amended to remove "wherein the scheduling".

In addressing claims 7, 14, and 21, the Action asserts that it is unclear if the claims are independent or dependent claims." Applicant respectfully submits that these claims are not indefinite.

Claim 7 recites ***one or more computer-readable media*** containing a computer executable program that performs a method as recited in claim 1. **Claim 14** recites ***one or more computer-readable media*** containing a computer executable program that performs a method as recited in claim 8. **Claim 21** recites ***a computer*** comprising one or more computer-readable media as recited in claim 15. Thus, claims 7 and 14 are directed to one or more computer-readable media and claim 21 is directed to a computer. As evidenced by the results of a cursory search of the PTO database, which uncovered a number of issued patents with claims written in the form asserted as being unclear by the Action, the Patent Office apparently agrees that the Action's rejected claim formats are not indefinite.

Specifically, consider U.S. Patent Nos. 6,725,262, 6,716,102, 6,674,918, and 6,433,266 exemplary claims of which are reproduced just below.

6,725,262

> 26. A computer-implemented method of synchronizing a configuration of resources on a plurality of computing devices comprising:

generating a set of lists that describes a configuration of resources that each of a plurality of computing devices should have in order to be synchronized with one another, the configuration of resources defining the content and the settings for each of the computing devices;

sending the set of lists to each of the computing devices;

receiving a response from one or more of the computing devices, each response requesting data that is needed in order to synchronize the configuration of resources for the corresponding computing device;

evaluating the response to determine what data is needed by a particular computing device to synchronize its resources; and

sending the data that is needed by the particular computing device to the computing device so that it can synchronize its resources.

33. ***One or more computer-readable media*** having computer-readable instructions thereon which, when executed by a computer, implement the method of claim 26.

In the above examples, claim 26 of 6,725,262 recites a computer-implemented method. Claim 33, which depends from claim 26, recites one or more computer-readable media with instructions which, when executed, implement the method of claim 26.

6,716,102

27. A method comprising:

receiving a request to save a game being executed by a gaming system;

saving a graphic representation of the saved game;

saving a descriptive name of the saved game; and

saving a date and time that the game was saved.

34. One or more computer-readable media comprising computer-executable instructions that, when executed, perform the method as recited in claim 27.

Here, claim 27 recites a method. Claim 34, which depends from claim 27, recites one or more computer-readable media with instructions which, when executed, perform the method of claim 27.

6,674,918

1. A computer-implemented method of synthesizing an image from at least two other images comprising:

acquiring a first image that serves as a color source for a resultant image which is to be formed;

acquiring a second image which serves as a perturbation source for the first image;

operating upon a plane that represents the first image by angularly perturbing vectors associated with the plane that represents the first image as a function of aspects of the second image to provide a perturbed image; and

applying an illumination model to the perturbed image to provide a resultant synthesized image.

9. One or more computer-readable media having computer-readable instructions thereon which, when executed by a computer, implement the method of claim 1.

Here, claim 1 recites a computer-implemented method. Claim 9, which depends from claim 1, recites one or more computer-readable media with instructions which, when executed, perform the method of claim 1.

6,433,266

1. One or more computer-readable media containing a computer program comprising:

a plurality of track objects representing different musical tracks;

a track manager that calls the track objects iteratively to play multiple track instances of at least a particular one of the musical tracks;

wherein the track manager indicates state information when calling the particular track object representing said particular musical track, the state information defining a current track state of a particular one of the multiple track instances of said particular musical track;

wherein said particular track object responds to the supplied state information by playing a portion of said particular musical track in accordance with the current track state defined by the indicated state information;

wherein the track manager keeps track of state information corresponding to said multiple track instances of said particular musical track.

5. A ***computer comprising the computer-readable media recited*** in claim 1.

In this example, claim 1 recites **one or more computer-readable** media. Claim 5, which depends from claim 1, recites **a computer comprising the computer-readable media recited** in claim 1.

In view of the above, it is respectfully submitted that there is nothing indefinite about pending claims 7, 14, and 21, and the Patent Office appears to agree with this assertion.

Accordingly, the 35 USC §112, second paragraph rejection of claims 7, 14, and 21 is improper and should be withdrawn.

35 USC §103(a) Rejections

Claims 1-4, 6-10, 12-23, and 25-29 stand rejected under 35 USC §103(a) as being unpatentable over U.S. Patent No. 6,308,279 to Toll et al ("Toll"). These rejections are traversed.

Claim 1 recites in part "scheduling one or more threads according to a predetermined periodic rate", "responsive to a determination that there are no threads to execute, deactivating at least one subset of components for a dynamic variable amount of time" and "the dynamic variable amount of time being independent of the predetermined periodic rate and being based on a sleep state of a set of threads in a sleep queue." In addressing claim 1, the Action asserts that these recited features are taught by col. 2, lines 22-34, and col. 3, lines 8-12 of Toll. Applicant disagrees. Nowhere does Toll, as modified by the Action, teach or suggest the recited features of claim 1.

Instead, Toll teaches a system for power mode transition in a multithreaded processor (Abstract). The portions of Toll cited by the Action (col. 2, lines 22-34, and col. 3, lines 8-12) teach that if even one thread is still executing, clock signals in a multi-threaded processor should not be completely turned off, internal clocks are turned off to save power, and the stop grant mode is used to clean-up operational state and drain queues before the system of Toll is transitioned into the deep sleep mode. Let's take a closer look at those portions of Toll that are cited by the Action.

Col. 2, lines 29-34 describe "[a]ll clock signals in the MT processor should not be turned off if even one thread is still in the active mode because the operations performed by that thread may still need synchronization. When every logical processor in a MT processor enter thread sleep state, the clocks on the MT processor may be turned off." Thus, Toll teaches that clock signals in a multi-threaded processor should not be completely turned off when even one thread is still executing

Referring now to col. 3, lines 8-12, Toll describes "[e]ventually, as a thread goes to sleep the micro-code associated with that thread stops running. When the threads in the MT processor are asleep, the hardware turns some of the internal clocks off to reduce the amount of power being used." Thus, Toll teaches that when threads in a MT are sleeping, internal clocks are turned off to save power.

Referring cited col. 3, lines 23-30, Toll describes "[w]hen the micro-code for all of the logical processors have executed, the MT processor may enter the stop grant mode. The chipset may then assert the signal on the sleep pin (SLP#), which places the processor in a sleep mode 130. After waiting an appropriate amount of time, the chipset may turn off the clocks by stopping a clock input signal to the processor (BCLK). This places the processor in a deep sleep power mode 140." Thus, Toll teaches that there are two modes, a stop grant mode and a deep sleep mode. With respect to the stop grant mode, Toll teaches at col. 1, lines 39-57 that the stop grant mode is used to clean-up operational state, drain queues, etc., before the system of Toll is transitioned into the deep sleep mode.

In view of the above, the cited portions of Toll teach: (a) that clock signals in a multi-threaded processor should not be completely turned off if even one thread is still executing (col. 2, lines 29-34); (b) when threads in a MT are sleeping, internal clocks are turned off to save power (col. 3, lines 8-12); and (c) the stop grant mode is used to clean-up operational state, drain queues, etc., before the system of Toll is transitioned into the deep sleep mode (col. 3, lines 23-30). These cited portions are completely silent with respect to "deactivating" anything for "a dynamic variable amount of

time" that is "independent of the predetermined periodic rate", the rate at which the threads are scheduled (i.e., "scheduling one or more threads according to a predetermined periodic rate"), and wherein the "dynamic variable amount of time" is "based on a sleep state of a set of threads in a sleep queue". Rather, Toll teaches at col. 1, lines 27-31 that resources are deactivated for subsequent re-activation as a function of received Input/Output and an external event such as "the opening of a lid on a laptop computer".

With respect to Toll's teaching that resources are deactivated for subsequent re-activation as a function of received Input/Output (col. 1, lines 27-31), Toll also teaches the following at col. 3, lines 12-18:

> *"[i]t should be noted that the core clocks may actually be left running, as in a debug mode, or the clock may be turned on to process a "snoop," in which case the processor will respond normally to the inquiry. When the processor senses a break event, it turns the internal clocks back on and returns to the active power 110 mode" With respect to the "break event", Toll teaches at col. 3, lines 4-8, that "[w]hen the MT processor samples the signal on the stop clock pin as 'asserted,' stop clock micro-code running in the MT processor will clean up the appropriate operational states and set up the correct 'break events,' or events that will cause the MT processor to wake up."*

These teachings that resources are deactivated for subsequent re-activation as a function of received Input/Output are completely silent with respect to "deactivating at least one subset of components for a dynamic variable amount of time" that is "independent of the predetermined periodic rate", the rate at which the threads are scheduled (i.e., "scheduling one or more threads according to a predetermined periodic rate").

This is especially since Toll is completely silent with respect to teaching any frequency of time at which the system of Toll will evaluate a queue of any type to determine whether there are any threads in that queue that need to be scheduled for execution. Because of this lack of teaching, it is likely that Toll evaluates any such queue for Input/Output threads for scheduling, as typically do all other conventional systems, based on a constant system tick rate. A system tick is the rate at which a hardware timer interrupt is generated and serviced by an operating system. When the timer fires, the operating system (OS) schedules a new thread for execution, if one is ready to be scheduled. Applicant has clearly described such conventional systems that evaluate thread scheduling queues at a predetermined constant system tick rate in the Background section of patent application. The Office is urged to review that section in view of these remarks.

With respect to Toll's second mode that teaches that a resource sleeps until occurrence of some external event such as "the opening of a lid on a laptop computer" (col. 1, lines 27-31), this does not teach "deactivating at least one subset of components for a dynamic variable amount of time" that is "based on a sleep state of a set of threads in a sleep queue", as claim 1 recites. Instead, such external events are just that, external, and not "based on a sleep state of a set of threads in a sleep queue", as claim 1 recites.

Accordingly, Toll's system, which deactivates resources until an I/O operation is received or until the occurrence of some external event such as "the opening of a lid on a laptop computer" (col. 1, lines 27-31), may never

"deactivating at least one subset of components for a dynamic variable amount of time" and "the dynamic variable amount of time being independent of the predetermined periodic rate and being based on a sleep state of a set of threads in a sleep queue", and wherein the "one or more threads" are scheduled "according to [the] predetermined periodic rate", as claim 1 recites.

With respect to the Action's modification to Toll, the Action modifies Toll to place sleeping threads into a sleep queue. This modification to Toll does not cure the above described deficiencies of Toll. Toll in view of the modification are completely silent with respect to deactivating anything for any "dynamic variable amount of time being independent of the predetermined periodic rate and being based on a sleep state of a set of threads in a sleep queue", wherein the "predetermined periodic rate" was used to "scheduling one or more threads". Thus, the combination of Toll with the Action's modification of a sleep queue does not present a prima facie case of obviousness over the recited features of claim 1.

Accordingly, the 35 USC §103(a) rejection of claim 1 over the cited combination is improper and should be withdrawn.

Claim 2 – 7 depend from claim 1 and are allowable over the cited combination solely by virtue of this dependency. Accordingly, and for this reason alone, the 35 USC §103(a) rejections of claims 2-7 over Toll is improper and should be withdrawn.

Additionally, claims 2-7 include additional subject matter that is not taught or suggested by the cited combination. For example, claim 6 recites

"setting a system timer to generate a notification at the predetermined periodic rate", "resetting the system timer to generate the notification after the dynamic variable amount of time has elapsed since the deactivating", "receiving the notification after the dynamic variable amount of time has elapsed since the deactivating", "responsive to the receiving: resetting the system timer to generate the notification at the predetermined periodic rate" and "activating the at least one subset of components." In addressing this feature, the Action concludes that it is taught by Toll at. col. 3, lines 15-34. This conclusion is unsupportable.

Let's take a look at the cited col. 3, lines 12-34:

> *"[i]t should be noted that the core clocks may actually be left running, as in a debug mode, or the clock may be turned on to process a 'snoop,' in which case the processor will respond normally to the inquiry. When the processor senses a break event, it turns the internal clocks back on and returns to the active power 110 mode.*
>
> *According to this particular embodiment of the present invention, when the stop clock micro-code executes for one logical processor in the MT processor, a stop grant acknowledge SBC is issued, including an identifier associated with that particular logical processor. When the micro-code for all of the logical processors have executed, the MT processor may enter the stop grant mode. The chipset may then assert the signal on the sleep pin (SLP#), which places the processor in a sleep mode 130. After waiting an appropriate amount of time, the chipset may turn off the clocks by stopping a clock input signal to the processor (BCLK). This places the processor in a deep sleep power mode 140 . As is also shown in FIG.* 1 , the processor may be *returned to the active power mode 110 by, for example, starting the BCLK, de-asserting the SLP# and de-asserting the STPCLK#".*

As clearly shown, this cited portion is completely silent with respect to any teaching of "setting a system timer to generate a notification at the predetermined periodic rate", as claim 6 recites. Recall that claim 1, from which claim 6 depends, recites "scheduling one or more threads according to a predetermined periodic rate". Instead, Toll merely indicates that the core clock in a system of Toll, responsive to an inquiry or external event, may wake a thread up before it was originally scheduled for waking. Nowhere does Toll indicate that the core clock is set to "generate a notification at the predetermined periodic rate", as claim 6 recites.

For this reason alone, Toll in view of the Action's modification to Toll, which adds a sleep queue to Toll, does not teach or suggest the features of claim 6.

Additionally, claim 6 recites "resetting the system timer to generate the notification after the dynamic variable amount of time has elapsed since the deactivating". As indicated above, Toll teaches that a thread may be woken up prior to its originally scheduled wake-up time. This is completely silent with respect to any teaching or suggestion of recites "resetting the system timer to generate the notification after the dynamic variable amount of time has elapsed since the deactivating".

For this additional reason, Toll in view of the Action's modification to Toll, which adds a sleep queue to Toll, does not teach or suggest the features of claim 6.

Additionally, claim 6 recites "receiving the notification after the dynamic variable amount of time has elapsed since the deactivating". As indicated above, Toll teaches that a thread may be woken up prior to its

originally scheduled wake-up time, not that the core timer sends a notification after the time that the thread was scheduled to wake-up had already passed. Thus, a system of Toll in view of the Actions modification to Toll may never "receiving the notification after the dynamic variable amount of time has elapsed since the deactivating", as claim 6 recites.

For this additional reason, Toll in view of the Action's modification to Toll, which adds a sleep queue to Toll, does not teach or suggest the features of claim 6.

Moreover, claim 6 also recites "responsive to the receiving: resetting the system timer to generate the notification at the predetermined periodic rate" and "activating the at least one subset of components." For the reasons already discussed, the system of Toll in view of the Action's modification may never perform this recited feature. Instead, Toll teaches that a thread may be made active (woken-up) prior to its scheduled wake-up time if an inquiry or external event such as the opening of a laptop computer lid is detected.

For this additional reason, Toll in view of the Action's modification to Toll, which adds a sleep queue to Toll, does not teach or suggest the features of claim 6.

Accordingly, and for each of the above reasons, the 35 USC §103(a) rejection of claim 6 over the cited combination is improper and should be withdrawn.

Claim 8 recites "scheduling one or more threads at a predetermined periodic rate", "responsive to a determination that there are no threads to execute, deactivating at least one subset of components for a dynamic

variable amount of time", "the dynamic variable amount of time being based on a sleep state of a set of threads in a sleep queue and independent of the predetermined periodic rate". For the reasons already discussed above with respect to claim 1, Toll in view of the Action's modification to Toll, which adds a sleep queue to Toll, does not teach or suggest the features of claim 8.

Accordingly, the 35 USC §103(a) rejection of claim 8 over the cited combination is improper and should be withdrawn.

Claims 9-14 depend from claim 8 and are allowable over the cited combination solely by virtue of this dependency. Accordingly, and for this reason alone, the 35 USC §103(a) rejections of claims 9-14 over the cited combination is improper and should be withdrawn.

Moreover, claim 12 includes additional features that are not taught or suggested by Toll in view of the Action's modification to Toll. Specifically, claim 12 recites "wherein the scheduling further comprises setting a system timer to the predetermined periodic rate, the predetermined periodic rate corresponding to a thread scheduling accuracy", and "wherein the deactivating further comprises resetting the system timer to generate a notification after the dynamic variable amount of time has elapsed since the deactivating." For the reasons already discussed above with respect to claim 6, Toll in view of the Action's modification to Toll, which adds a sleep queue to Toll, does not teach or suggest the features of claim 12.

Accordingly, and for this additional reason, the 35 USC §103(a) rejection of claim 12 over the cited combination is improper and should be withdrawn.

In another example, claim 13 recites "wherein the deactivating further comprises resetting a system timer to generate a notification after the dynamic variable amount of time has elapsed, the dynamic variable amount of time being a maximum amount of time that a thread can yield to other threads before needing to be scheduled for execution", and "wherein the activating further comprises resetting the system timer to the predetermined periodic rate to provide substantial thread scheduling accuracy." For the reasons already discussed above with respect to claim 6, Toll in view of the Action's modification to Toll, which adds a sleep queue to Toll, does not teach or suggest the features of claim 13.

Accordingly, and for this additional reason, the 35 USC §103(a) rejection of claim 13 over the cited combination is improper and should be withdrawn.

Claim 15 recites "determining at a periodic rate whether or not there are any threads to execute", and "responsive to a determination that there are no threads to execute, deactivating at least one subset of components for a dynamic variable amount of time, the at least one subset being selected from a group of components comprising the one or more of the program modules and one or more of the hardware elements, the dynamic variable amount of time being independent of the periodic rate, the dynamic variable amount of time being based on a sleep state of a set of threads in a sleep queue." For the reasons already discussed above with respect to claim 1, Toll in view of the Action's modification to Toll, which adds a sleep queue to Toll, does not teach or suggest the features of claim 15.

Accordingly, the 35 USC §103(a) rejection of claim 15 over the cited combination is improper and should be withdrawn.

Claims 16-21 depend from claim 15 and are allowable over Toll solely by virtue of this dependency.

Accordingly, and for this reason alone, the 35 USC §103(a) rejections of claims 16-21 over the cited combination is improper and should be withdrawn.

Moreover, claim 19 includes additional features that are not taught or suggested by Toll in view of the Action's modification to Toll. Specifically, claim 19 recites "in the deactivating, configuring a system timer to send a first timer interrupt after the dynamic variable amount of time has elapsed, the dynamic variable amount of time being a maximum amount of time that a first thread can yield to a second thread before the first thread needs to be executed", and "responsive to receiving the first timer interrupt: (a) configuring the system timer to send a second timer interrupt at the periodic rate" and "(b) activating the deactivated at least one subset of components to determine if there are any threads to execute." At least for the reasons already discussed above with respect to claim 6, Toll in view of the Action's modification to Toll, which adds a sleep queue to Toll, does not teach or suggest the features of claim 19.

Accordingly, and for this additional reason, the 35 USC §103(a) rejection of claim 19 over the cited combination is improper and should be withdrawn.

Claim 22 recites in part "a hardware abstraction layer (HAL)". In addressing this feature, the Action points to col. 1, lines 11-24 and lines 32-

46, to conclude that this feature is taught by Toll. This conclusion is unsupportable. Let's take a look at the cited portions of Toll. Col. 1, lines 11-24 recite:

> *"The amount of power used by the processor will impact, for example, how long a battery in a mobile computer will last. Designers, therefore, have attempted to limit the power used by a processor.*
>
> *Even when not performing mathematical operations, the generation and distribution of internal clock signals that synchronize the processor's operation will consume a considerable amount of power. To save power, a processor may be designed to operate in a reduced power state when inactive. In the reduced power state, all but a few internal clocks are turned off, which saves power and may extend the life of a battery."*

Clearly, nowhere does the above cited portion of Toll teach or suggest the claimed "hardware abstraction layer" of claim 22. Additionally, Toll at col. 1, lines 21-46 recite:

> *"To aid in energy efficient computing, in some implementations the processor is placed into an even lower power state referred to as a "deep sleep" power mode. The deep sleep mode may be entered, for example, by stopping a clock input signal to the processor after the processor has entered the sleep power mode. This allows the processor to maintain the operational state of elements in the chip, but only draws power equivalent to the processor's leakage current.*
>
> *With highly complex processors, such as out-of-order processors, some internal "clean-up" may be desired before the internal clocks are disabled. Such clean up is typically performed by micro-code which, for example, cleans up the operational state, drains queues, puts the processor to sleep and waits for an event, or "alarm," that marks the end of the hibernation."*

As in the first cited portion of Toll, nowhere does the immediately above cited portion of Toll teach or suggest the claimed "hardware abstraction layer" of claim 22. Thus, the system of Toll may never include this claimed feature of claim 22. For this reason alone, Toll in view of the Action's modification to Toll, which adds a sleep queue to Toll, does not teach or suggest the features of claim 22.

Additionally, claim 22 also recites "scheduling threads for execution at a periodic time interval", and "wherein the HAL, responsive to the determining, comprises computer-executable instructions for deactivating, for a dynamic variable amount of time, at least one subset of components selected from a group of components comprising the scheduler, the hardware elements, the one or more operating system program modules, and the application program modules, the dynamic variable amount of time being independent of the periodic time interval and being based on a sleep state of a set of threads in a sleep queue." For the reasons already discussed above with respect to claim 1, Toll in view of the Action's modification to Toll, which adds a sleep queue to Toll, does not teach or suggest these cited portions of claim 122.

Accordingly, for each of the above reasons, the 35 USC §103(a) rejection of claim 22 over the cited combination is improper and should be withdrawn.

Claims 23-29 depend from claim 22 and are allowable over the cited combination solely by virtue of this dependency.

Accordingly, and for this reason alone, the 35 USC §103(a) rejections of claims 23-29 over the cited combination is improper and should be withdrawn.

Moreover, claim 29 includes additional features that are not taught or suggested by Toll in view of the Action's modification to Toll. Specifically, claim 29 recites "receiving a notification in response to an external event, the external event not being a system timer event, responsive to receipt of the notification, the HAL processing the notification in a manner that the scheduler remains deactivated for the dynamic variable amount of time." At least for the reasons already discussed above with respect to claim 6, Toll in view of the Action's modification to Toll, which adds a sleep queue to Toll, does not teach or suggest the features of claim 29.

Accordingly, and for this additional reason, the 35 USC §103(a) rejection of claim 29 over the cited combination is improper and should be withdrawn.

As an additional matter, if claim 29 is again rejected on the same basis, it is respectfully requested for the Office to particularly point out where Toll teaches or suggests "the HAL processing the notification in a manner that the scheduler remains deactivated for the dynamic variable amount of time", as claim 29 recites.

Conclusion

Pending claims 1-29 are in condition for allowance, and action to that end is respectfully requested. Should any issue remain that prevents allowance of the application, the Office is encouraged to contact the undersigned prior or issuance of a subsequent Office action.

Respectfully submitted,

Dated: January 03, 2005 By: Brian Hart

Brian G. Hart
Reg. No. 44, 421
(509) 324-9256